

(19)



JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: 10333929 A

(43) Date of publication of application: 18 . 12 . 98

(51) Int. Cl

G06F 9/46

G06F 15/16

(21) Application number: 10070882

(71) Applicant: FUJITSU LTD

(22) Date of filing: 19 . 03 . 98

(72) Inventor: WAKABAYASHI HIROHIKO

(30) Priority: 31 . 03 . 97 JP 09 79261

(54) JOB EXECUTION SYSTEM AND RECORDING MEDIUM

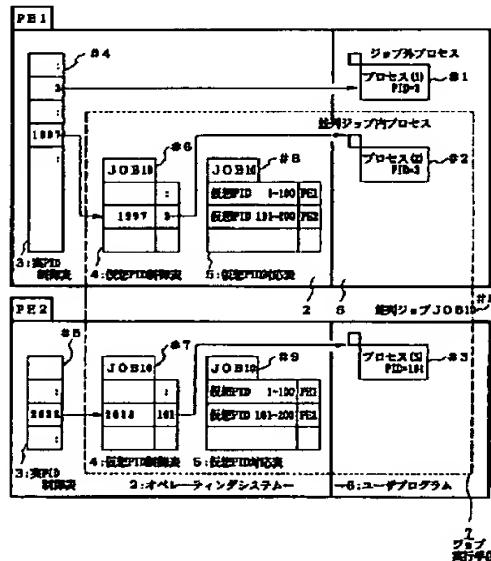
is eliminated without the reservation of the process ID.

COPYRIGHT: (C)1998,JPO

(57) Abstract:

PROBLEM TO BE SOLVED: To prevent the collision of processes by registering virtual processes respectively allocated to the plural processes in a virtual PID control table, allocating the empty real PID of a real PID control table, registering it in correspondence and making the respective processes communicate with each other by using the virtual process and perform processings within a job.

SOLUTION: Plural processors PE1 and PE2 are constituted of the real PID control table 3 and a job execution means 7, etc. Further, the job execution means 7 provides the virtual PID control table 4 for the respective jobs and registers unique virtual process IDs respectively allocated to the plural processes for executing the jobs in the virtual PID control table 4. Then, the empty real PID of the real PID control table 3 is allocated and registered in correspondence and the respective processes communicate with each other by using the virtual process IDs and execute the processing. Thus, the collision between the process IDs



(19)日本国特許庁 (JP)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平10-333929

(43)公開日 平成10年(1998)12月18日

(51)Int.Cl.⁶

G 0 6 F 9/46
15/16

識別記号

3 6 0

F I

G 0 6 F 9/46
15/16

3 6 0 B
4 2 0 J

審査請求 未請求 請求項の数5 OL (全9頁)

(21)出願番号 特願平10-70882

(22)出願日 平成10年(1998)3月19日

(31)優先権主張番号 特願平9-79261

(32)優先日 平9(1997)3月31日

(33)優先権主張国 日本 (JP)

(71)出願人 000005223

富士通株式会社

神奈川県川崎市中原区上小田中4丁目1番
1号

(72)発明者 若林 裕彦

神奈川県川崎市中原区上小田中4丁目1番
1号 富士通株式会社内

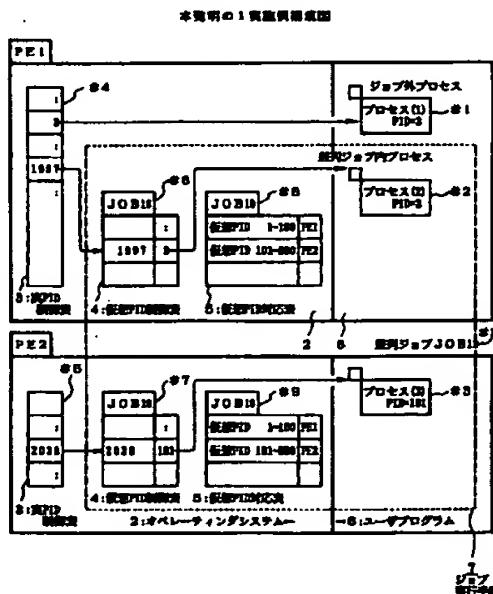
(74)代理人 弁理士 岡田 守弘

(54)【発明の名称】 ジョブ実行システムおよび記録媒体

(57)【要約】

【課題】 本発明は、ジョブ実行システムおよび記録媒体に関し、ジョブ毎に任意の一意の仮想プロセスIDの割り当てを実現し、凍結したジョブを解凍したときにプロセスIDの予約なしにプロセスIDの衝突を無くして即時にジョブ内のプロセス間でデータのやりとりを行う処理の再開を可能にすることを目的とする。

【解決手段】 複数のプロセッサ上でそれぞれ処理を実行するプロセスにそれぞれ割り当てた一意の実プロセスIDを登録して管理する実PID制御表と、複数のプロセッサ上でジョブを実行する複数のプロセスに対してジョブ内でそれぞれ割り当てた一意の仮想プロセスIDを、一意の実プロセスIDにそれぞれ対応づけて登録して管理する仮想PID制御表とを各プロセッサに設け、ジョブ内では各プロセスが仮想プロセスIDを用いて相互に通信し、処理を実行する手段を備えるように構成する。



【特許請求の範囲】

【請求項1】複数プロセッサ上で任意数のプロセスによってジョブを実行するジョブ実行システムにおいて、複数のプロセッサ上でそれぞれ処理を実行するプロセスにそれぞれ割り当てた一意の実プロセスIDを登録して管理する実PID制御表と、複数のプロセッサ上でジョブを実行する複数のプロセスに対して当該ジョブ内でそれぞれ割り当てた一意の仮想プロセスIDを、上記一意の実プロセスIDにそれぞれ対応づけて登録して管理する仮想PID制御表とを各プロセッサに設け、上記ジョブ内では各プロセスが仮想プロセスIDを用いて相互に通信し、処理を実行する手段を備えたことを特徴とするジョブ実行システム。

【請求項2】各プロセッサ内にジョブ毎に仮想プロセスIDがいずれのプロセッサに存在するかを登録する仮想PID対応表を設け、

ジョブ内の仮想プロセスIDがいずれのプロセッサに存在するかを判別してそのプロセッサにデータを送信してジョブ内の各プロセスが相互にデータの送受信を行うことを特徴とする請求項1記載のジョブ実行システム。

【請求項3】ジョブの凍結指示に対応して、指示されたジョブの上記仮想PID制御表とジョブ内のプロセス、あるいは更に上記仮想PID対応表を退避して保存することを特徴とする請求項1あるいは請求項2記載のジョブ実行システム。

【請求項4】ジョブの解凍指示に対応して、上記保存しておいたジョブの仮想PID制御表とジョブ内のプロセス、あるいは更に上記仮想PID対応表を復元した後、当該仮想PID制御表中に登録されている仮想プロセスIDに対して上記実PID制御表の空の実プロセスIDを割り当てて登録した後、ジョブ内の各プロセスが処理を再開することを特徴とする請求項1ないし請求項3記載のいずれかのジョブ実行システム。

【請求項5】複数のプロセッサ上でそれぞれ処理を実行するプロセスにそれぞれ割り当てた一意の実プロセスIDを実PID制御表に登録して管理する手段と、複数のプロセッサ上でジョブを実行する複数のプロセスに対して当該ジョブ内でそれぞれ割り当てた一意の仮想プロセスIDを、上記一意の実プロセスIDにそれぞれ対応づけて仮想PID制御表に登録して管理する手段と、上記ジョブ内では各プロセスが仮想プロセスIDを用いて相互に通信し、処理を実行する手段と、して機能させるプログラムを記録したコンピュータ読み取可能な記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、複数プロセッサ上で任意数のプロセスによってジョブを実行するジョブ実

行システムおよび記録媒体に関するものである。

【0002】

【従来の技術】従来、複数のPE(プロセッサ)間にまたがるプロセスの集合体によって実行するいわゆる並列ジョブを任意の時点で中断させ、その状態をファイルなどに保存して凍結し、システムの IPLなどを経た後に、再びファイルから凍結したジョブを読み出し、当該ジョブの実行を継続するジョブ凍結・解凍機能がある。

10 【0003】このジョブ凍結・解凍の際に問題となるのは次の点である。ジョブに属するプロセス中では、そのプロセスの識別子(プロセスID)を使用して、ジョブ内のプロセス間のデータのやりとりを行うが、ジョブ凍結の際には、プログラム空間内にそのプロセスIDの情報が残っているため、ジョブ解凍時には、ジョブ内のプロセスについては、以前に持っていたプロセスIDを保証した形で、ジョブを再現してプロセスIDの衝突を回避する必要がある。

20 【0004】このとき、当該ジョブ外のプロセスで、そのプロセスIDと同一のプロセスIDが既に使用中である場合には、そのプロセスの終了を待たなければ、ジョブを再開できない。

【0005】これを解決するため、プロセスの凍結時にプロセスIDに関する情報を格納しておき、システム再起動時に格納情報をもとにプロセスIDを予約し、ジョブを解凍して再現し、他のジョブが非所望に当該プロセスIDを使用することによる待ちを無くし、即時に予約しておいた以前のプロセスIDを用いてプロセス間のデータのやりとりを開始して処理を再開するようになっていた。

30 【0006】

【発明が解決しようとする課題】上述したように、従来のジョブ凍結・解凍機能では、解凍時のプロセスIDの衝突を避けるために、システムの起動時にプロセスIDを予め予約しておく必要があり、このために解凍するジョブに応じた予約のための格納情報を確保し、システム起動時に予約して確保する必要が生じる。

40 【0007】このような従来のジョブ凍結・解凍機能では、予めプロセスIDの予約が必要であり、凍結された任意のジョブを解凍できないという制約が生じる問題があった。

【0008】また、並列プログラムでは、ジョブを構成するプログラムがPEの数だけ存在するため、多数のプロセスIDを予約しておく必要があり、その分だけシステム中で常に使用できないプロセスIDが存在し、他のプロセスのプロセスIDの探索範囲を圧迫してしまうという問題も発生する。特にジョブ凍結・解凍機能が要求される大規模なジョブ程、並列度が高く、多数のプロセスを必要とする傾向があり、極めて多数のプロセスIDが使用できなくなってしまうという問題がある。

50 【0009】本発明は、これらの問題を解決するため、

一意な実プロセスIDを管理する実PID制御表の他にジョブ内の仮想プロセスIDを一意に管理する仮想PID制御表を設けて実プロセスIDと仮想プロセスIDとの対応づけを行い、ジョブ毎に任意の一意の仮想プロセスIDの割り当てを実現し、凍結したジョブを解凍したときにプロセスIDの予約なしにプロセスIDの衝突を無くして即時にジョブ内のプロセス間でデータのやりとりを行う処理の再開を可能にすることを目的としている。

【0010】

【課題を解決するための手段】図1を参照して課題を解決するための手段を説明する。図1において、プロセッサPE1、PE2は、複数のプロセッサであって、プロセスを生成してジョブを実行するためのものであり、ここでは、実PID制御表3、およびジョブ実行手段7などから構成されるものである。

【0011】実PID制御表3は、プロセスに一意に割り当てた実プロセスIDを登録して管理するものである。ジョブ実行手段7は、複数プロセッサ上で任意個のプロセスによってジョブを実行するものであって、図示のように、仮想PID制御表4、仮想PID対応表5、および複数のプロセスから構成されるものである。

【0012】仮想PID制御表4は、ジョブ内のプロセスに一意に割り当てた仮想プロセスIDに対応づけて実プロセスIDを登録して管理するものである。仮想PID対応表5は、仮想プロセスIDがいずれのプロセッサにあるかを登録したものである。

【0013】次に、動作を説明する。ジョブ毎に仮想PID制御表4を設けてジョブを実行する複数のプロセスにそれぞれ割り当てた一意の仮想プロセスIDを当該仮想PID制御表4に登録すると共に実PID制御表3の空の実PIDを割り当てて対応づけて登録し、ジョブ内では各プロセスが仮想プロセスIDを用いて相互に通信し、処理を実行するようにしている。

【0014】この際、各プロセッサ内に仮想プロセスIDがいずれのプロセッサに存在するかを登録する仮想PID対応表5を設け、ジョブ内の仮想プロセスIDがいずれのプロセッサに存在するかを判別してそのプロセッサにデータを送信してジョブ内の各プロセスが相互にデータの送受信を行い、処理を実行するようにしている。

【0015】また、ジョブの凍結指示に対応して、指示されたジョブの仮想PID制御表4とジョブ内のプロセス、あるいは更に仮想PID対応表5を退避して保存するようにしている。

【0016】また、ジョブの解凍指示に対応して、保存しておいたジョブの仮想PID制御表4とジョブ内のプロセス、あるいは更に仮想PID対応表5を復元した後、仮想PID制御表4中に登録されている仮想プロセスIDに対して実PID制御表3の空の実プロセスIDを割り当てて登録した後、ジョブ内の各プロセスが処理

を再開するようにしている。

【0017】従って、一意な実プロセスIDを管理する実PID制御表3の他にジョブ内の仮想プロセスIDを一意に管理する仮想PID制御表4を設けて実プロセスIDと仮想プロセスIDとの対応づけを行うことにより、ジョブ毎に任意の一意の仮想プロセスIDの割り当てを実現し、凍結したジョブを解凍したときにプロセスIDの予約なしにプロセスIDの衝突を無くして即時にジョブ内のプロセス間でデータのやりとりを行う処理を

10 再開することが可能となる。

【0018】

【発明の実施の形態】次に、図1から図5を用いて本発明の実施の形態および動作を順次詳細に説明する。ここで、記録媒体から読み出したプログラムあるいは外部記憶装置であるハードディスク装置などから読み出したプログラム、またはセンタの外部記憶装置から読み出して回線を介して転送を受けたプログラムを主記憶にローディングして起動し、以下に説明する各種処理を行うようしている。

20 【0019】図1は、本発明の1実施例構成図を示す。図1において、PE1、PE2は、複数のプロセッサのうちの2つを表したものであって、図示のように、オペレーティングシステム2およびユーザプログラム6の部分から構成されるものである。

【0020】オペレーティングシステム(OS)2は、全体を統括制御するものであって、ここでは、図示のように、実プロセスID(実PID)および仮想プロセスID(仮想PID)を割り当ておよび管理するための実PID制御表3およびジョブ実行手段7を構成するうち30 の仮想PID制御表4、仮想PID対応表5からなるものである。

【0021】実PID制御表3は、ユーザプログラム6の部分に創成したプロセスに一意に割り当て実プロセスID(実PID)を登録して管理するものである。仮想PID制御表4は、ジョブ内でユーザプログラム6の部分に創成したプロセスに一意に割り当て仮想プロセスID(仮想PID)を登録して管理するものである。

【0022】仮想PID対応表5は、仮想プロセスIDの所定範囲がいずれのプロセッサ(PE)にあるかを登録したものである(図5参照)。プロセスは、ユーザプログラム6の部分に創成して各種処理を実行するものであって、相互にデータを通信などするために一意のプロセスID(実PIDあるいは仮想PID)を割り当てたものである。

【0023】次に、図2のフローチャートに示す順序に従い、図1の構成のもとでプロセスID(実PIDおよび仮想PID)を割り当てるときの手順を詳細に説明する。図2は、本発明の動作説明フローチャートを示す。ここで、PE1、PE2は、図1のプロセッサであるPE1、PE2である。

【0024】図2において、S1、S2は、PE1、PE2がOSのIPLを行う。ここでは、OSのIPLの過程でPE1、PE2は、プロセスの実プロセスIDを一意に割り当てて登録して管理するための実PID制御表#4、#5の作成を行う。

【0025】S3は、PE1に並列ジョブ#10が投入される。この延長で、PE1、PE2を使用する並列ジョブ#10が実行されることとなる。S4、S5は、S3で並列ジョブ#10が投入されると、ジョブ毎の仮想PID制御表#6、#7の作成を行う。これにより、ジョブ毎に仮想プロセスIDと実プロセスIDの対応関係を登録して管理する仮想PID制御表#6、#7の作成が行われたこととなる。

【0026】S6、S7は、マスターのプロセッサPE1が仮想PID対応表#8を作成して各PE、ここではPE2への配布を行い当該PE2で仮想PID対応表#9を作成する。この仮想PID対応表#8、#9により、各PEにおいて仮想プロセスIDとPEとの対応関係を認識し、仮想プロセスIDをもとにいずれのPEにデータを送信すればよいかを判別することが可能となる。ここでは、仮想PID対応表#8、#9には

仮想プロセスID PE

001~100 PE1

101~200 PE2

と登録されているので、仮想プロセスIDが001~100まではPE1に存在し、101~200まではPE2に存在することが判明する

S8は、PE1で親プロセス#2を生成する。そして、親プロセス#2は、S9で子プロセス#3をPE2に生成する。

【0027】S10は、S8で親プロセスを生成したことに対応して、PE1では、実PID制御表#4の空の実プロセスIDとして「1997」を割り当てて登録する。S11は、S10と同様に、S9で子プロセスを生成したことに対応して、PE2では、実PID制御表#5の空の実プロセスIDとして「2038」を割り当てて登録する。

【0028】S12は、空き仮想プロセスID(3)の割り当てを行うと共に、実プロセスID(1997)と仮想プロセスID(3)を対応づけて仮想PID制御表#6への登録を行う。これにより、仮想PID制御表#6には、実プロセスID(3)と仮想プロセスID(1997)との対応関係が登録されたこととなる。

【0029】S13は、S12と同様に、仮想プロセスID(101)の割り当てを行うと共に、実プロセスID(2038)と仮想プロセスID(101)を対応づけて仮想PID制御表#7への登録を行う。これにより、仮想PID制御表#7には、実プロセスID(2038)と仮想プロセスID(101)との対応関係が登録されたこととなる。

【0030】以上の手順によって、実プロセスIDと仮想プロセスIDを制御するための内部データ構造が構築されたこととなる。そして、ユーザプログラムへは、S12、S13で割り当てられた仮想プロセスIDが通知され、ユーザプログラムは自分、あるいは親プロセス#2、子プロセス#3のプロセスIDを認識する。ここでは、親プロセス#2は「3」(実際は実プロセスID、1997)、子プロセス#3は「101」(実際は実プロセスID、2038)と認識する。

10 【0031】次に、図3のフローチャートに示す順序に従い、図1の構成のもとで、図2の手順によって実プロセスIDと仮想プロセスIDを制御するための内部データ構造が構築された状態で、ジョブを実行するときの動作を詳細に説明する。

【0032】図3は、本発明の動作説明フローチャートを示す。ここで、PE1、PE2は、図1のプロセッサであるPE1、PE2である。図3において、S21は、プロセス#2(親プロセス)からPID=101へデータを送る。

20 【0033】S22は、PE1のオペレーティングシステム2が仮想PID対応表#8をみて通信先PEを決定する。ここで、PID=101は、PE2と決定する。S23は、S22でPE2と決定されたので、PE2へ並列ジョブ#10のPID=101のプロセスへデータを送る。これは、S22で通信先PEがPE2と決定されたので、PE1と同一ジョブ#10の配下の仮想プロセスID(101)のプロセスに対して、データを送信する。

30 【0034】S24は、S23で送信されたデータを受信したPE2のオペレーティングシステム2が、並列ジョブ#10の仮想PID制御表#3を見てプロセス#3へデータを渡す。そして、ジョブを実行する。

【0035】以上によって、図1のPE1の並列ジョブ#10内のプロセス#2からPID=101へデータ送信したことに対応して、仮想PID対応表#8を参照してPE2にデータが渡され、PE2内で更に仮想PID制御表#7を参照して並列ジョブ#11内のプロセス#3にデータが渡されたこととなる。これらにより、仮想PID対応表を参照して該当するPEにデータを送信し、次に当該PE内で仮想PID制御表を参照して該当するプロセスにデータを渡し、ジョブ内で異なるPE内のプロセスにデータを送信できたこととなる。

40 【0036】次に、図4のフローチャートに示す順序に従い、図1の構成のもとで、ジョブの凍結および解凍について詳細に説明する。図4は、本発明の動作説明フローチャートを示す。

【0037】図4において、S31は、ジョブ凍結指示を受信する。これは、OSがシステム管理者あるいはユーザからジョブ凍結指示を受信する。そして、ジョブ実行中の各PEにジョブ凍結指示を通知する。

【0038】S32は、PE2がS31で通知したジョブ凍結指示を受信する。S33、S34は、各PE内で並列ジョブ#10の仮想PID対応表#8、#9、仮想PID制御表#6、#7とプロセス#2、#3をそれぞれファイルに退避して保存する。

【0039】S35は、ジョブ凍結完了する。以上によって、ジョブ凍結指示を受信したときに、凍結指示を受けたジョブを実行している各PEは、当該ジョブの仮想PID対応表、仮想PID制御表、およびプロセスをそれぞれファイルに退避して保存する。

【0040】S40は、ジョブ解凍を開始する。S41は、ジョブ解凍指示を受信する。これは、OSがシステム管理者あるいはユーザからジョブ解凍指示を受信する。そして、ジョブを割り付ける各PEにジョブ解凍指示を通知する。

【0041】S42は、PE2がS41で通知したジョブ解凍指示を受信する。S43、S44は、各PE内で並列ジョブ#10の仮想PID対応表#8、#9、仮想PID制御表#6、#7の復元を行う。そして、更にプロセス#2、#3の情報を合わせてPE1、PE2に復元する。

【0042】S45、S46は、実PID制御表#4、#5の空プロセスIDの再割り当てと仮想PID制御表#6、#7への登録を行う。これにより、実PID制御表#4、#5の空の実プロセスIDを割り当てて、仮想PID制御表#6、#7に実プロセスIDと仮想プロセスIDとの対応づけが登録され、実プロセスIDと仮想プロセスIDとの対応関係が決定されることとなる。

【0043】S47、S48は、プロセス#2、#3の復元を行い、ジョブの実行を再開する。以上のように、仮想PID対応表、仮想PID制御表、プロセスの情報を復元した後、実PID制御表から空の実プロセスIDを割り当ててこの実プロセスIDを仮想PID制御表に登録して仮想プロセスIDとの対応関係を決定した後、プロセスを復元してジョブの実行を再開することが可能となる。このように、実プロセスIDの割り当てとして、実PID制御表を参照して空の実プロセスIDを仮想プロセスIDに対応づけて仮想PID制御表に登録しているため、確実に仮想プロセスIDに実プロセスIDを割り当ることができ、従来の他のプロセスが凍結時と同一の実プロセスIDを使用してその終了を待つまで待機するという事態の発生を無くし、迅速に実プロセスIDを仮想プロセスIDに割り当ててジョブの実行を再開することが可能となった。

【0044】図5は、本発明の仮想PID対応表例を示す。この仮想PID対応表5は、仮想プロセスIDの範囲に対応づけてプロセッサIDを登録したものである。ここでは、仮想プロセスIDの範囲が001～100の場合に、プロセッサPE1と登録し、仮想プロセスIDが001～100のプロセスは、プロセッサPE1上で動作するものである旨が判明する。従って、仮想プロセスIDが001～100の範囲内の仮想プロセスIDへのデータ送信依頼の場合には、データをPE1に送信すればよい。そして、PE1は、受信したデータについて、仮想PID制御表4を参照して該当する仮想プロセスIDのプロセスに渡す。

【0045】

【発明の効果】以上説明したように、本発明によれば、一意な実プロセスIDを管理する実PID制御表3の他にジョブ内の仮想プロセスIDを一意に管理する仮想PID制御表4を設けて実プロセスIDと仮想プロセスIDとの対応づけを行う構成を採用しているため、ジョブ毎に任意の一意の仮想プロセスIDの割り当てを実現し、凍結したジョブを解凍したときに従来のプロセスIDの予約なしにプロセスIDの衝突を無くして即時にジョブ内のプロセス間でデータのやりとりを行う処理を再開することができる。また、ユーザプログラムの構成（プロセスの構成）を意識することなく、解凍時にプロセスIDの再割り当てを行うときに自動的にプロセスIDの衝突を回避してジョブの再開を確実かつ迅速に行い、処理の続行することが可能となる。

【図面の簡単な説明】

【図1】本発明の1実施例構成図である。

30 30 【図2】本発明の動作説明フローチャート（プロセスIDの割り当て）である。

【図3】本発明の動作説明フローチャート（仮想プロセスIDを使用したデータ送受信）である。

【図4】本発明の動作説明フローチャート（ジョブ凍結、ジョブ解凍時のプロセスIDの割り当て）である。

【図5】本発明の仮想PID対応表例である。

【符号の説明】

2：オペレーティングシステム（OS）

3：実PID制御表

40 4：仮想PID制御表

5：仮想PID対応表

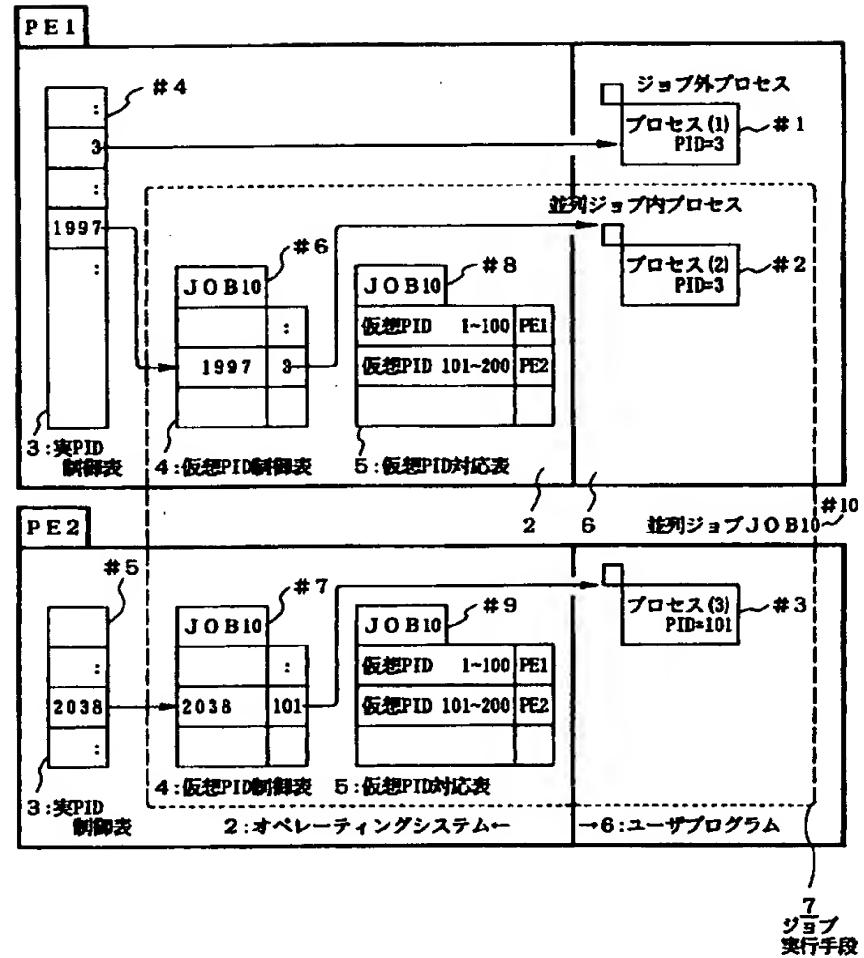
6：ユーザプログラム

7：ジョブ実行手段

PE、PE1、PE2：プロセッサ

【図1】

本発明の1実施例構成図

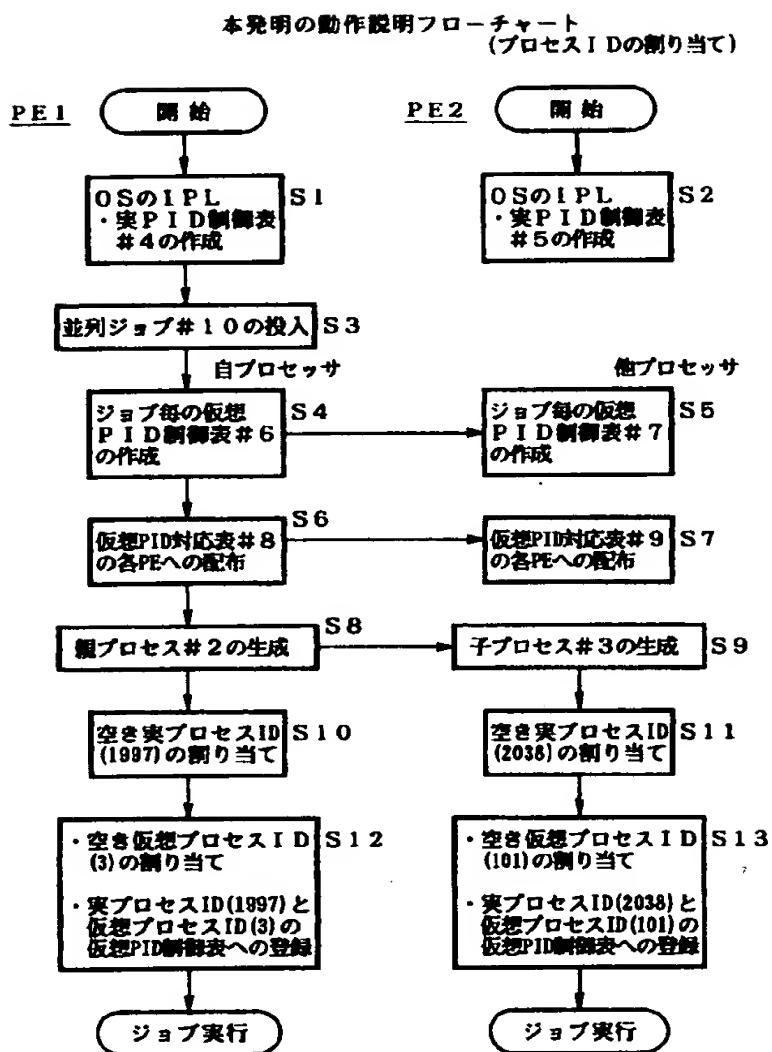


【図5】

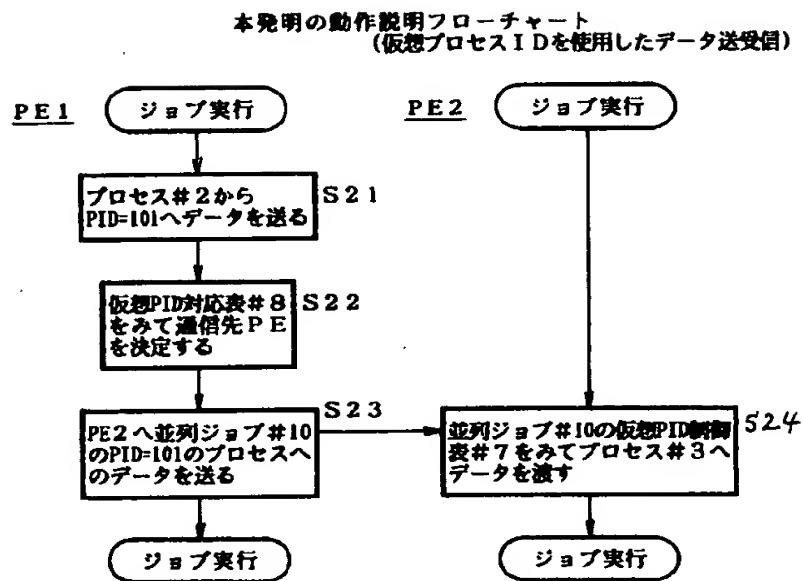
本発明の仮想PID対応表

仮想プロセスID	プロセッサID	8
001~100	PE1	
101~200	PE2	

【図2】



【図3】



【図4】

